

SHAPA: AN INTERACTIVE SOFTWARE TOOL FOR PROTOCOL ANALYSIS APPLIED TO AIRCREW COMMUNICATIONS AND WORKLOAD

Jeffrey M. James, Penelope M. Sanderson, and Karen S. Seidler
Department of Mechanical and Industrial Engineering,
University of Illinois at Urbana-Champaign
Urbana, IL 61801

ABSTRACT

As modern transport environments become increasingly complex, issues such as crew communication, interaction with automation, and workload management have become crucial. Much research is being focused on holistic aspects of social and cognitive behavior, such as the strategies used to handle workload, the flow of information, the scheduling of tasks, the verbal and non-verbal interactions between crew members. Traditional laboratory performance measures no longer sufficiently meet the needs of researchers addressing these issues. However observational techniques are better equipped to capture the type of data needed and to build models of the requisite level of sophistication. Presented here is SHAPA, an interactive software tool for performing both verbal and non-verbal protocol analysis. It has been developed with the idea of affording the researcher the closest possible degree of engagement with protocol data. The researcher can configure SHAPA to encode protocols using any theoretical framework or encoding vocabulary that is desired. SHAPA allows protocol analysis to be performed at any level of analysis, and it supplies a wide variety of tools for data aggregation, manipulation. The output generated by SHAPA can be used alone or in combination with other performance variables to get a rich picture of the influences on sequences of verbal or non-verbal behavior.

INTRODUCTION

Current Research Issues in Transport Environments

Today's cockpit environment is a challenging one. Crew members have to handle complex information that arrives through a variety of different channels: information management systems, visual displays, ATC, other crew members and so on. In addition, new on-board systems are continually being developed that need to be evaluated and refined in the context of piloting tasks.

Automation introduces a variety of issues. In many ways, it acts as an extra crew member. It is an entity that not only demands attention but is a source of information. As such, it contributes to the coordinative and information complexity in the cockpit. How do on-board computers affect how decisions are made and how information flows between crew members? Does automation decrease the amount of information overtly communicated, possibly resulting in misunderstanding? A computer may disrupt more accustomed exchanges between crew members by providing total or partial solutions to problems, or by interruption. Monitoring automated systems may itself produce workload. Alternatively, pilots may experience "underload", and feel removed from the basic piloting task. Does the operator trust the computer? Can the computer be used effectively to offload workload? Who takes responsibility for the allocation, and how does variation in workload between operators affect overall crew performance? These issues are being addressed by various researchers (Foushee, 1984; Wiener, 1985)

Because there are moments of distinct overload and quite long periods of underload, the management of workload becomes crucial. The individual must manage his or her own workload and, where relevant, do so in the context of how workload is allocated in a group of people. If a computer is present, its role in adding or taking away workload must be evaluated.

Hart (1988) has argued that overload and underload exist on a continuum. If overloaded, operators may defer or even shed tasks, and may choose tasks to perform on a different basis than usual. Hart argues that many tasks in a mission are discrete in

nature and have "windows of opportunity" within which they can be completed. Given this, operators actually have a reasonable amount of discretion in how they organize their time, and may develop strategies for doing so. Workload then becomes a function not only of initial conditions such as task requirements, interface and operator resources, and operator experience, but also of how the operator assesses and reacts to the situation as it unfolds, and as he/she believes it will be in the future. Accordingly, the influence of conscious, strategic factors on workload and performance is now receiving a lot of attention.

Once it is accepted that the operator manages his or her workload strategically and creatively, research questions become less molecular and more molar. The determinants of workload will now lie in answers to questions such as: What does the operator know about the current situation? Is the operator happy with his or her level of performance? Does the operator have a particular strategy--or short-cut--for achieving a goal in the present situation? How does the operator manage an uneven level of workload? How does the operator prioritize tasks when they vary in terms of priority, time for completion, inherent cognitive or motor difficulty, and probability of being successfully completed within the time available? How will an operator's strategy change with time pressure?

It is clear that molecular information processing measures of performance such as reaction time and RMS error fail to answer such questions. New conceptual and methodological tools will be required. These tools will have to be sensitive to global, conscious, and deliberate aspects of behavior. In particular, it will be very important to be able to categorize behavior and detect patterns. Only on this basis can new models of workload be developed.

Observational Techniques

Many of the issues described above are not amenable to controlled experimentation, and traditional performance measures such as reaction time do not convey sufficient information to build the types of conceptual models needed. Observational techniques are far more suitable as they can provide ways of collecting data in a way that does not artificially

constrain behavior and they provide various ways of analyzing data gathered in naturalistic environments. For example, non-verbal protocols can be collected of crew actions and gestures. In addition, there is much to be gained from eliciting conscious knowledge about strategies, tactics and other concerns from those we observe in such environments. Verbal protocols can be collected, where appropriate, to give an idea of the cognitive processes and strategies used. It is argued that many methods of data reduction used in ethology will be useful in analyzing these two sources of data. Both these can be analyzed alongside the more traditional performance measures.

There have been debates about the validity of verbal protocol data. Ericsson and Simon (1984) have argued that it should be possible to treat verbal data like any other sort of data. They argue that all data processing requires transformation from an initial observation to a form in which theories can be tested: "the cognitive processes that generate verbalizations are a subset of the cognitive processes that generate any kind of recordable response or behavior." (p. 9). As with any other data, researchers should be aware of the strengths and limitations of verbal data. For example, verbal data is most likely to be valid when a subject has been thinking about the task verbally and the information remains in short term memory.

If verbal data is to be a powerful tool in theory development and evaluation, then it is imperative that it can be subjected to various aggregation and analysis routines in order to achieve succinct representations of the information it contains. For example, it can be subjected to the types of data reduction techniques currently being used for observational data. Verbalization, after all, is one of the most important elements of the human behavioral repertoire, and can be used as a vehicle for organizing, directing and evaluating action towards a goal.

The other type of protocol analysis is non-verbal protocol analysis. Non-verbal protocol analysis stretches to all types of observations of serial behavior such as sequences of facial expressions, gestures, play behavior, and non-verbal communication (Scherer and Ekman, 1982). Such observational techniques are usually not nearly as intrusive, or potentially intrusive, as verbal protocol techniques, because the subject is not necessarily aware of being observed. Rigorous analytical techniques are seemingly better applied to the more objective, less self-conscious nature of non-verbal protocols. However, verbal and non-verbal data might be equally successfully analyzed with ethological data reduction techniques.

When used together, verbal protocol analysis and non-verbal observational techniques are capable of capturing much of the richness of a situation being examined. However, both techniques involve detailed transcription and analysis, which are very time-consuming. One of the principal goals of the work discussed herein is to develop methodological tools that will allow such data to be analyzed easily, and sound models to be developed quickly. The SHAPA environment, to be described, is a first step towards this. SHAPA is designed to afford researchers more direct engagement with a large, immensely rich, but hitherto relatively undigestible, body of data.

Purpose of SHAPA

Protocol analysis is notoriously difficult and time-consuming to perform. The time required for analysis of a protocol has been estimated to be an order of magnitude greater than the time required to actually record the protocol! This means that too often researchers use protocols as *anecdotal* support for theories or points being made, without building any kind of a statistical case on the basis of verbal data for the assertions that it is supporting. Any tools that can shorten this process and help

verbal data be treated as any other sort of data, where appropriate, are invaluable.

It has been clear that what is needed is a highly general, interactive protocol analysis environment where the human encoder still makes the high-level judgments required, but the computer takes the burden for much of the "hack" work. SHAPA is a coordinated, interactive protocol analysis environment where researchers can encode a wide variety of data according to categories of their own choosing, in the context of a model or theory of their choosing. *Thus SHAPA is to protocol data what a spreadsheet program is to numerical data, or what a word processor is to text: it is intelligent about the sorts of things a researcher might want to do with verbal or non-verbal protocols, while being blind to particular domains, contexts, or theories.* SHAPA should allow a researcher to see more quickly the patterns in various types of sequential behavior. This should expedite data analysis and model development.

SHAPA allows the researcher to carry out the steps for protocol analysis suggested by Ericsson and Simon (1980, 1984) as well as others. Verbal statements, or segments of non-verbal behavior, can be encoded as *atomic formulae* from the predicate calculus. Predicate calculus is an AI-based language for expressing propositions and their relations in a standard format (see Charniak and McDermott, 1986). SHAPA does not assume that researchers always wish to use the entire predicate calculus to encode raw protocols. It merely uses the simpler of its conventions as an encoding syntax.

SHAPA provides a set of tools for identifying strategies and information flow from complex verbal and non-verbal protocols. At present, SHAPA best handles protocols from individual subjects, but we are embarking on a project to make it better handle protocols from multiple sources in parallel. SHAPA will continue to evolve as models of the impact of automation, crew communications and workload-management strategies develop. However it should, in its turn, accelerate the development of useful models in these areas. The use of such a tool speeds the development of useful models of strategic and interpersonal factors in cockpits, control rooms, or operations centers. It allows protocols to be analyzed at various levels of abstraction, allowing tests of different models of performance. Additionally, better models of workload can be built. We will be able better to identify and classify situations where performance will be threatened, providing predictive tools for designers of instrumentation and decision support systems.

PROTOCOL ENCODING USING SHAPA

The steps required for the more detailed encoding of protocols will now be enumerated, and the role of SHAPA described where relevant. This treatment deals mostly with the analysis of verbal data; however, it is easy to see how the technique could be extended to nonverbal data such as motor and performance data. In practice, the steps outlined here are seldom done in such a linear sequence: there is usually much back-tracking, adjustment and correction as protocol analysis gets under way. This is fully supported in SHAPA.

1. Task Analysis

Even before the protocol is taken, the researcher should understand the problem space of the particular task at hand. The important system states of the task environment should be identified, along with the important operators (actions). This is equivalent to identifying the "problem space" (Newell and Simon, 1972). The researcher should start to develop a

descriptive language for task states, perceptual states and operators. However, the development of the descriptive language can be helped by SHAPA.

It may only become clear what the syntax and vocabulary should be once the encoding has started. There can be a "bootstrapping" cycle here: the best descriptive language may only become obvious once the encoding begins. This is particularly true during the first few protocols in a set.

2. Transcription and Segmentation

The process of eliciting a verbal protocol from a subject will not be discussed here (the reader might consult the Appendix in Ericsson and Simon, 1984, for a short discussion of this). The stream of verbalization or behavior needs to be transcribed from audio or video tape and broken into segments for encoding. Researchers may also want to include time codes and symbolic representations of pauses in speech or behavior.

Segmentation involves breaking the stream of verbalization into sentences, clauses or phrases that express one idea and that can be encoded with one predicate (see 3. and 4.). Alternatively, segmentation can be performed where there are pauses in verbalization. The segments may be syntactically and/or semantically distinct, but must be sufficiently cognitively distinct to be encoded in a predicate.

3. Determine Encoding Vocabulary and Notation

Encoding vocabulary. As mentioned above, an encoding vocabulary, or descriptive language, needs to be developed. The basic idea is that when handling any sequence of behavior or verbalization, decisions have to be made about (1) what aspects of the situation are worthy of being noted, (2) what aspects of the situation must be distinguished from each other, and (3) what aspects are sufficiently similar to be classed together. When determining the encoding vocabulary, an important goal should be to reduce the variability in the natural language, or unconstrained behavior, to essential propositions which retain the semantics of the situation *in the context of the theoretical disposition of the encoders*.

SHAPA is relatively theoretically "agnostic" and has been designed to handle a wide variety of representations for problem-solving tasks. Researchers need to generate a working set of important distinctions they want to make about the behavior or utterances in the protocol record. Each of these distinctions can be qualified by what is, for all practical purposes, an infinite number of qualifiers or arguments. Obviously, however, an encoder wants to reduce a protocol to its "essentials", according to a theoretical viewpoint, so will choose qualifiers and arguments as parsimoniously as possible.

Encoding notation. Once the encoding vocabulary has been decided upon, there needs to be a standard, convenient notation for encoding--this will increase the reliability of the analysis. SHAPA uses something rather like the "atomic formula" of predicate calculus as a general encoding notation (Charniak and McDermott, 1986). The atomic formula consists of a *predicate* (also called an *operator*) and its *arguments* (also called its *terms*). Predicates can be verbs or nouns that represent the verbal or non-verbal activity that is of interest, or the propositional content of an utterance. The arguments qualify the predicate and provide details about the current situation. This fundamental representational syntax does not mean that SHAPA requires that thought or action be modeled according to the entire predicate calculus, even though it might sometimes be appropriate. It is simply a general and flexible way of representing the content of an utterance.

1. The predicate MONITOR might need to be qualified by (1) what is being monitored and (2) why it is being monitored.
2. The predicate COMMAND might need to be qualified by (1) who gives the command, (2) to whom it is directed, (3) the content of the command, and (4) the directness or indirectness of the command.
3. The predicate STATE might reflect comments about current system state, and include (1) the parameter being discussed and (2) its value.

The encoding vocabulary should handle as many of the verbalizations as possible, including queries, exclamations, and so on. Similarly, "place-holding arguments" should be chosen that indicate the syntax for the predicate arguments. Thus the *canonical* form of these predicates would be as follows:

1. MONITOR(<WHAT>,<WHY>)
2. COMMAND(<BY>,<TO>,<CONTENT>,<DIRECTNESS>)
3. STATE(<PARAMETER>,<VALUE>)

The place-holding arguments are surrounded by brackets to indicate that they have not yet been replaced by a *constant* based on the content of the raw protocol.

4. Perform Encoding.

This is where the SHAPA environment is used to its fullest extent. The SHAPA interface is similar to a full-screen editor, with certain constraints imposed by the nature of the raw protocol and protocol encoding files and the need to preserve their integrity. The encoder can move through the file much in the way one would in a word processor. SHAPA provides various screen format options for how the raw protocol and protocol encoding files can be displayed on the screen segment by segment. SHAPA's default layout alternates between lines of raw protocol and lines of encoding as shown in Figure 1.

Encoding individual segments. When the encoder wishes to encode a segment, he or she moves the cursor to the protocol encoding file line associated with that segment. He or she decides upon the appropriate predicate for the segment, and enters an abbreviated form of the predicate name (e.g., G for GOAL for C for COMMAND). SHAPA then displays the appropriate syntactical form of the predicate with the place-holding arguments--GOAL(<PARAMETER>,<VALUE>)--in the associated line of the protocol encoding file. The encoder then uses the cursor keys to move through the displayed predicate to replace place-holding arguments with standardized representations of the content of the protocol segment. For

File	Encode	Predicates	Search	Layout	Report
Line 1	Col 1	MEALPREP.1->MEALENC.1			Insert
1	I'm going to plan a dinner party for six people	GOAL(PLAN(dinner for 6, , , ,			
2	two of those people will be myself and Bill	LIST(quests,)			
3	two will be my parents,	()			
4	and two will be a couple that I know from Chicago	()			
5	The sort of dinner party that I want is one with three courses.	GOAL(3-courses, ,)			
6	I'm going to have a hard day's work before hand	()			
7	so I'm going to try to do it all in two hours	CONSTRAINT(time, ,)			
8	So what I have to do is plan the different courses that I want,	GOAL(PLAN(courses, , , ,)			
9	the three courses,	()			
10	work out what I need to buy.	GOAL(PLAN(purchases, , , ,)			
11	I'll go to the supermarket	PLAN(visit-supermarket, ,)			
Syntax: GOAL(<WHAT>,<VALENCE>,<CHARACTER>)					
Alt-A Toggle Active					

Figure 1. SHAPA encoding screen with alternating line layout

4. Collection of Predicate Instances

For some applications, researchers will be interested in collecting together segments that have been encoded with the same predicate. For example, in a study of cockpit communication it is important to examine information inquiries among crew members in order to determine information flow. All segments requesting information could be encoded as INQUIRY statements. These statements needed to be collected together so that a graphical representation of the crew's flow of communication could be developed.

SHAPA will report selected encoding lines on the basis of the predicate name or on the basis of a particular constant used as one of the arguments for that predicate. Thus one might collect all INQUIRY statements regardless of the way their arguments have been encoded. Alternatively, one might collect all INQUIRY statements where the constant for the place-holding argument <SPEAKER> could be, for example, captain. This would allow one to examine all inquiries made by the captain of the aircraft.

5. Cross-Reliability

In the context of verbal protocol analysis, reliability is the tendency for one encoding of a protocol to be similar to another, whether they be separate encodings by one person or by different people. Verbal protocol analysis cannot have any pretensions to validity unless at some level it is reliable. Good agreement between encoders suggests a clear, unambiguous protocol with respect to the encoding vocabulary chosen, but does not guarantee that what has been said bears a faithful relationship to the mental processes involved in the task in question. Thus reliability does not guarantee validity.

SHAPA provides a reliability cross-check between the predicates used in two encodings of the one raw protocol file. It can compare encodings performed with the same set of predicates or two different sets of predicates. When comparing two files, SHAPA sets up a matrix that has the predicates used in the first encoding as the rows, and the predicates used in the second encoding as the columns. Each encoded segment is represented as a tally in the appropriate cell of the matrix. When the rows and columns represent the same set of predicates, then two encodings with good reliability would reveal a large number of observations on the left diagonal. With this matrix approach, predicates resulting in high and low reliability can easily be distinguished. Another feature of this approach is that the predicates on the rows and columns need not be the same. The matrix-based reliability cross-check allows a researcher to see which predicates might be considered to refer to the same thing, and which predicates have only partially overlapping meanings.

6. Transition Matrices

An important way of capturing patterns in the protocol record is to look at the sequencing of behavior. Transition frequency matrices can be constructed from which a variety of analyses can be performed (van Hooff, 1982). Transition analyses help to

First Order Transition Matrix		Successor							
Predecessor		J	C	G	P	L	EC	M	
JUDGE		5	2	1	5	1	.	2	
CONSTRAINT		1	1	2	1	1	.	1	
GOAL		1	3	1	5	1	.	.	
PLAN		5	1	3	5	5	.	1	
LIST		2	.	1	4	.	1	.	
E-COMMENT		.	.	1	
MEANS		2	.	1	.	1	.	.	

Figure 3. First order transition matrix

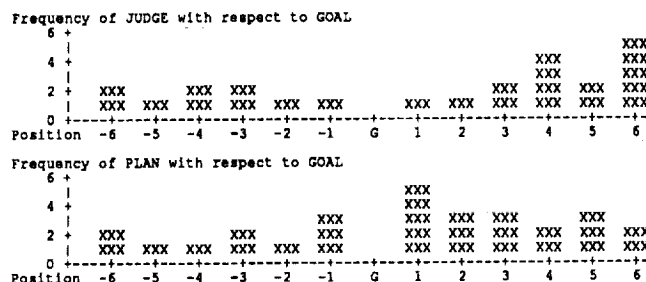


Figure 4. Lag sequential analysis

determine whether a verbal segment or piece of behavior is generally strongly influenced by the behavior before it; that is, whether there are dependencies in the data matrix. Such analyses allow the researcher to detect subsystems of verbalization or behavior that serve a specific function.

The analysis may thus reveal habitual or stereotyped patterns of behavior, such as GOAL-->STATUS-->PLAN. Such patterns may indicate elemental functional *next* such as planning a course of action or anticipating future activity. It is then up to the researcher to determine how much the regularity of the pattern is due to environmental constraints and how much to human preferences and strategies, and to draw conclusions appropriate to the hypotheses being tested. A first order transition matrix generated by SHAPA is shown in Figure 3. SHAPA also provides second and third order transition matrices which are potentially one and two orders of magnitude larger.

7. Lag Sequential Analysis

In some domains, whether with verbal or non-verbal data, there may be a tendency for one type of utterance or activity to precede or follow another at a certain remove. For instance, in crew communication a response may tend always to come between two and four statements following an inquiry. This general tendency might be missed by transitional matrix analyses, where strict sequences of predicates are used. Lag sequential analysis allows these more vague patterns and dependencies in the sequence of protocol segments to be discerned (Douglas and Tweed, 1979). Figure 4 contains two graphs of a lag sequential analysis report generated by SHAPA. Separate graphs are generated for each available predicate.

8. Frequency of Cycles

The final analysis has been adopted from Fisher (1988), and is similar to various ethological techniques for finding regularities in behavior sequences. "Fisher's Cycles", as we will call them, provide a report of actually occurring sequences of predicates rather than formally defined sequences as in transition analyses. It is a powerful heuristic for identifying the patterns in the data that relies upon human rather than machine pattern recognition ability. It also avoids the intensive computation often needed to identify and compare patterns.

More details about SHAPA can be found in Sanderson, James, and Seidler (1989).

CONCLUSION

As a piece of software, SHAPA is in the cognitive engineering tradition of exploiting human pattern-recognition abilities to amplify the user's "intelligence" (Norman and Draper, 1986; Woods and Roth, 1988). SHAPA is designed to facilitate the initial determination of appropriate vocabularies for encoding and to speed up the encoding and analysis of a series of files once an encoding scheme has been established. In this way, researchers can encode a greater number of protocols and generate measures of verbal or non-verbal behavior that can more quickly become amenable to conventional inferential statistics, or other pattern-recognition techniques. SHAPA is designed to engender a feeling of *direct engagement* with the protocol data and to make the manipulation of data as direct as possible.

With experience in analyzing verbal protocols, the researcher will have educated himself or herself in what to look for in a protocol—the patterns of verbalizations or actions and the evidence for theoretically important distinctions. Future protocols should then be much easier to analyze because the patterns should be easier to discern. Ideally, at this point the researcher should have the conceptual tools to classify subjects' conscious strategies on the basis of a smaller, more diagnostic sample of behavior. One may finally reach the point where a well-structured questionnaire, interview or behavioral test allows the subject to classify themselves reliably with respect to the categories important for the research at hand (see Sanderson, 1989, for an example on a small scale).

Protocol analysis and observational techniques support the kind of conceptual model-building that is sufficient to support design decisions. They often suggest hypotheses that might be tested in further observation or in more controlled experimental manipulations. One of the byproducts of the proposed work is that the conceptual advances offered by model building in the SHAPA encourage more focussed observational techniques, or experimental 'spin-offs' that can be performed in more controlled environments. In other words, SHAPA should help researchers learn what to look for, and how to measure it.

As further requirements of protocol analysis come to light, SHAPA will be developed to incorporate them as far as possible. In the future, SHAPA will be developed on the Macintosh so as to include more "visualization aids" where researchers can explore different ways of graphically representing protocol data to show patterns in their data. SHAPA will also be streamlined so that it more comfortably handles protocol data from multiple interacting agents, such as groups of people or a person interacting with a computer.

Major developments of SHAPA are currently being determined. Development of SHAPA will include integrating time into the protocol analysis. Time stamping of encoded lines will allow coordination with video protocols through a VCR interface as well as with continuous and discrete status variables taken from the environment. Features and analyses for integrating multiple encodings of the same protocol will be considered. Analysis of results across protocols will allow for a more holistic approach once simple protocol analysis through SHAPA has established. Further filtering and isolation of status variables, predicates, and arguments, and representation of data patterns in graphical form, will allow for greater visualization of the encoding.

Developing such a coordinated data analysis environment will be a large job, but not a conceptually difficult one once the basic needs and constraints had been identified. In the meantime, we feel that SHAPA represents a significant step towards solving some of the practical problems of performing protocol analysis, such as swift access to the data and speed of manipulation. It also enriches the types of conclusions that can be drawn by importing analyses from the non-verbal domain.

REFERENCES

- Card, S., Moran, T., and Newell, A. (1983). *The Psychology of Human-Computer Interaction*. Hillsdale, NJ: Erlbaum.
- Charniak, E. and McDermott, D. (1986). *Introduction to Artificial Intelligence*, Reading, MA: Addison-Wesley.
- Douglas, J.M. and Tweed, R.L. (1979). Analysing the patterning of a sequence of discrete behavioral events. *Animal Behavior*, 27, 1236-1252.
- Ericsson, K.A. and Simon, H.A. (1980). Verbal reports as data. *Psychological Review*, 87, 215-251.
- Ericsson, K.A. and Simon, H.A. (1984). *Protocol Analysis*. Cambridge, MA: MIT Press.
- Fisher, C. (1988). Advancing the study of programming with computer-aided protocol analysis. In G. Olson, E. Soloway, and S. Sheppard (Eds.), *Empirical Studies of Programmers, 1987 Workshop*. Norwood, NJ: Ablex Publishing Corporation.
- Foushee, H. C. (1984). Dyads and Triads at 35,000 feet: Factors affecting group process and air crew performance. *American Psychologist*, 39, 885-893.
- Hart, S. (1988). Presentation given to NASA contractees at Human Factors Society Meeting, Anaheim, CA. October 28, 1988.
- Mitchell, C. and Miller, R.A. (1986). A discrete control model of operator function: A methodology for information display design. *IEEE Transactions on Systems, Man and Cybernetics, SMC-16*, 343-357.
- Newell, A. and Simon, H.A. (1972). *Human Problem Solving*. Englewood Cliffs, NJ: Prentice-Hall.
- Norman, D.A. and Draper, S.W. (1986). *User-Centred System Design: New Perspectives on Human-Computer Interaction*. Hillsdale, NJ: Erlbaum.
- Sanderson, P. M. (1989). Verbalizable knowledge and skilled task performance: Association, dissociation, and mental models. *Journal of Experimental Psychology: Learning, Memory and Cognition*, 15, 729-747.
- Sanderson, P.M., James, J. and Seidler, K. (1989). *SHAPA: A software environment for verbal and nonverbal protocol analysis*. (Tech. Report EPRL-89-09). Urbana-Champaign: Engineering Psychology Research Laboratory, University of Illinois.
- Van Hooff, J.A.R.A.M. (1982). Categories and sequences of behavior: methods of description and analysis. In K.R. Scherer and P. Ekman (Eds.) *Handbook of Methods in Nonverbal Behavior Research*. Cambridge: Cambridge University Press.
- Wiener, E. L. (1985). Beyond the Sterile Cockpit. *Human Factors*, 27, 75-90.
- Woods, D.D. and Roth, E. (1988). Cognitive Systems Engineering. In M. Helander (Ed.) *Handbook of Human-Computer Interaction*. New York: Wiley.